


```

RRRRRRRR      MM      MM      000000      CCCCCCCC      RRRRRRRR      EEEEEEEEEEE      CCCCCCCC      000000      MM      MM
RRRRRRRR      MM      MM      000000      CCCCCCCC      RRRRRRRR      EEEEEEEEEEE      CCCCCCCC      000000      MM      MM
RR      RR      MMMM      MMMM      00      00      CC      RRR      RR      EE      CC      00      00      MMMM      MMMM
RR      RR      MMMM      MMMM      00      00      CC      RRR      RR      EE      CC      00      00      MMMM      MMMM
RR      RR      MM      MM      00      0000      CC      RRR      RR      EE      CC      00      00      MM      MM
RR      RR      MM      MM      00      0000      CC      RRR      RR      EE      CC      00      00      MM      MM
RRRRRRRR      MM      MM      00      00      00      CC      RRRRRRRR      EEEEEEEEEEE      CC      00      00      MM      MM
RRRRRRRR      MM      MM      00      00      00      CC      RRRRRRRR      EEEEEEEEEEE      CC      00      00      MM      MM
RR      RR      MM      MM      0000      00      CC      RR      RR      EE      CC      00      00      MM      MM
RR      RR      MM      MM      0000      00      CC      RR      RR      EE      CC      00      00      MM      MM
RR      RR      MM      MM      00      00      CC      RR      RR      EE      CC      00      00      MM      MM
RR      RR      MM      MM      00      00      CC      RR      RR      EE      CC      00      00      MM      MM
RR      RR      MM      MM      000000      CCCCCCCC      RR      RR      EEEEEEEEEEE      CCCCCCCC      000000      MM      MM
RR      RR      MM      MM      000000      CCCCCCCC      RR      RR      EEEEEEEEEEE      CCCCCCCC      000000      MM      MM

```

```

LL      IIIIII      SSSSSSSS
LL      IIIIII      SSSSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SSSSSS
LL      II      SSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LLLLLLLLLLLL      IIIIII      SSSSSSSS
LLLLLLLLLLLL      IIIIII      SSSSSSSS

```

```

.....
.....
.....
.....

```


(2) 156
(3) 253
(9) 662
(10) 765
(11) 835

DECLARATIONS
RM\$CREATECOM - PERFORM CREATE FILE FUNCTION
RM\$INI_CRE_RJR
Routines to journal CREATE attributes.
RM\$JNL_CREATE


```

0000 1      $BEGIN RMOCRECOM,000,RM$RMS0,<COMMON CREATE FILE>
0000 2
0000 3
0000 4      *****
0000 5      *
0000 6      *  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 7      *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 8      *  ALL RIGHTS RESERVED.
0000 9      *
0000 10     *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 11     *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 12     *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 13     *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 14     *  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 15     *  TRANSFERRED.
0000 16     *
0000 17     *  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 18     *  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 19     *  CORPORATION.
0000 20     *
0000 21     *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 22     *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 23     *
0000 24     *
0000 25     *****
0000 26
0000 27     ++
0000 28     Facility: rms32
0000 29
0000 30     Abstract:      this module performs the create file fcp function.
0000 31
0000 32     Environment:
0000 33                  star processor running starlet exec.
0000 34
0000 35     Author: L F Laverdure,      Creation Date: 11-MAY-1977
0000 36
0000 37     Modified By:
0000 38
0000 39     V03-028 RAS0309      Ron Schaefer      15-Jun-1984
0000 40                  Check for errors from RM$CREACC_SET1.
0000 41
0000 42     V03-027 JWT0175      Jim Teague        10-Apr-1984
0000 43                  Move ATR page deallocation code.
0000 44
0000 45     V03-026 DGB0033      Donald G. Blair    22-Mar-1984
0000 46                  Fill in XAB$L_ACLSTS during call to rm$xab_scan.
0000 47
0000 48     V03-025 JWT0166      Jim Teague        20-Mar-1984
0000 49                  Use dynamically-allocated scratch page for accumulating
0000 50                  ATRs for QIOs.
0000 51
0000 52     V03-024 DGB0007      Donald G. Blair    04-Mar-1984
0000 53                  Make changes related to the way we call the ACP in order
0000 54                  to support access mode protected files.
0000 55
0000 56     V03-023 JWT0148      Jim Teague        14-Dec-1983
0000 57                  Enforce RU settings (RU, ONLY_RU, NEVER_RU).

```


0000	58	:	
0000	59	:	
0000	60	:	V03-022 LMP0133 L. Mark Pilant, 3-Aug-1983 14:53
0000	61	:	Get default protection from PCB instead of PI space. Also
0000	62	:	don't supply the protection attribute unless explicitly
0000	63	:	given in a PROtection XAB.
0000	64	:	
0000	65	:	V03-021 RAS0164 Ron Schaefer 27-Jun-1983
0000	66	:	Fix 5 broken branches to RMOJOURNAL routines.
0000	67	:	
0000	68	:	V03-020 KPL0006 Peter Lieberwirth 22-Jun-1983
0000	69	:	Add support to journal expiration date on file creation.
0000	70	:	Add a mask to tell recovery which attributes have been
0000	71	:	journalled.
0000	72	:	
0000	73	:	V03-019 KPL0005 Peter Lieberwirth 16-Jun-1983
0000	74	:	Fix bug in MOV3 of FIB to create AT entry.
0000	75	:	
0000	76	:	V03-018 TSK0001 Tamar Krichevsky 15-Jun-1983
0000	77	:	Fix broken branches into journaling psect.
0000	78	:	
0000	79	:	V03-017 KPL0004 Peter Lieberwirth 3-Jun-1983
0000	80	:	Fix journaling \$CREATE attribute handling. Use correct
0000	81	:	RJR FIB entry.
0000	82	:	
0000	83	:	V03-016 KPL0003 Peter Lieberwirth 30-May-1983
0000	84	:	Fix error path bugs introduced in V03-015.
0000	85	:	
0000	86	:	V03-015 KPL0002 Peter Lieberwirth 16-May-1983
0000	87	:	Add initial support for journaling \$CREATEs. Clean
0000	88	:	up some old code, also. Changes for robust RJR format.
0000	89	:	
0000	90	:	V03-014 RAS0153 Ron Schaefer 2-May-1983
0000	91	:	Delete reference to \$XABACEDEF missed by RAS0148.
0000	92	:	
0000	93	:	V03-013 RAS0148 Ron Schaefer 26-Apr-1983
0000	94	:	Add initial support for extended XABPRO.
0000	95	:	
0000	96	:	V03-012 JWH0216 Jeffrey W. Horn 14-Apr-1983
0000	97	:	Re-organize journaling support so that we always
0000	98	:	write journal names and the Id-ACE after the file
0000	99	:	is accessed.
0000	100	:	
0000	101	:	V03-011 JWH0196 Jeffrey W. Horn 18-Mar-1983
0000	102	:	Add support for XABACE.
0000	103	:	
0000	104	:	V03-010 SHZ0001 Stephen H. Zalewski 21-Dec-1982
0000	105	:	Store the Files-11 hbk and ebk in different fields in the
0000	106	:	ifb than we keep the swapped hbk and ebk.
0000	107	:	
0000	108	:	V03-009 JWH0161 Jeffrey W. Horn 21-Dec-1982
0000	109	:	Reset some FIB fields when performing IOS_MODIFY
0000	110	:	to write journal name.
0000	111	:	
0000	112	:	V03-008 MCN0001 Maria del C. Nasr 08-Dec-1982
0000	113	:	If the FIB alignment option is RFI with a file id of
0000	114	:	zero, change it to ANY so that we do not get a file
		:	not found error from the ACP.

0000 115 :
0000 116 :
0000 117 :
0000 118 :
0000 119 :
0000 120 :
0000 121 :
0000 122 :
0000 123 :
0000 124 :
0000 125 :
0000 126 :
0000 127 :
0000 128 :
0000 129 :
0000 130 :
0000 131 :
0000 132 :
0000 133 :
0000 134 :
0000 135 :
0000 136 :
0000 137 :
0000 138 :
0000 139 :
0000 140 :
0000 141 :
0000 142 :
0000 143 :
0000 144 :
0000 145 :
0000 146 :
0000 147 :
0000 148 :
0000 149 :
0000 150 :
0000 151 :
0000 152 :
0000 153 :
0000 154 :

V03-007 ACG0306 Andrew C. Goldstein, 13-Dec-1982 14:57
Remove obsolete file header symbols

V03-006 JWH0103 Jeffrey W. Horn 20-Sep-1982
Remove RMSASSJNL, RMSMAPJNL calls. This code
has been moved to RM\$SETEBK in RM0ACCESS.

V03-005 JWH0110 Jeffrey W. Horn 29-Sep-1982
Fix typos in V03-004.

V03-004 JWH0109 Jeffrey W. Horn 29-Sep-1982
Fix problems with CIF logic in V03-003.

V03-003 JWH0002 Jeffrey W. Horn 31-Aug-1982
Fix problems CIF logic in V03-001.
Add support for Recovery Unit Journals.

V03-002 KBT0203 Keith B. Thompson 23-Aug-1982
Reorganize psects

V03-001 JWH0001 Jeffrey W. Horn 02-Jul-1982
Put in journaling support.

V02-028 KPL0001 Peter Lieberwirth 28-Dec-1981
Do a better job deleting the file after errors returned by
RM\$SETEBK by getting the DID from the FWA FIBBUF instead
of the user NAM block. The problem is there may be no NAM
block, so a dangling directory entry can result.

V02-027 MCN0007 Maria del C. Nasr 12-May-1981
Define new symbol for old length of backup date and time XAB.

V02-026 JAK0048 J A KRYCKA 25-SEP-1980
Move network specific create code to RMSOCREAT and avoid
spurious setting of NAM\$V_HIGHVER and NAM\$V_LOWVER bits.

V025 REFORMAT D M WALP 24-JUL-1980


```

0000 156      .SBTTL  DECLARATIONS
0000 157
0000 158      ;
0000 159      ; Include Files:
0000 160      ;
0000 161
0000 162      ;
0000 163      ; Macros:
0000 164      ;
0000 165
0000 166      $IODEF
0000 167      $SSDEF
0000 168      $RJRDEF
0000 169      $RJBDEF
0000 170      $BDBDEF
0000 171      $CJFDEF
0000 172      $IMPDEF      ; impure area definitions
0000 173      $FABDEF
0000 174      $NAMDEF
0000 175      $FCHDEF
0000 176      $FIBDEF
0000 177      $IFBDEF
0000 178      $ATRDEF
0000 179      $DEVDEF
0000 180      $FWADEF
0000 181      $RMSDEF
0000 182      $XABALLDEF
0000 183      $XABDATDEF
0000 184      $XABFHCDEF
0000 185      $XABPRODEF
0000 186      $XABRDTDEF
0000 187      $XABJNLDEF
0000 188      $RUCBDEF
0000 189      $PCBDEF
0000 190
0000 191      ;
0000 192      ; Equated Symbols:
0000 193      ;
0000 194
00000020 0000 195      FOP = $AB$L_FOP*8      ; bit offset to fop
0000 196
0000 197      ;
0000 198      ; Offsets and sizes for local table entries.
0000 199      ;
00000000 0000 200      JNL$W_ATTR TYPE = 0      ; attribute type
00000004 0000 201      JNL$L_ACTION RTN = 4      ; address of action routine
00000008 0000 202      JNL$C_ENT_SIZE = 8      ; size of table entry
00000008 0000 203      ATR$S_ENT_SIZE = 8      ; size of ACP attribute
0000 204
0000 205      ;
0000 206      ; Own Storage:
0000 207      ;
0000 208      ;
0000 209      ; initial xab processing arguments
0000 210      ;
0000 211
0000 212  CRE_XAB_ARGS:

```

```

00'20 14 0000 213 .BYTE XAB$C_ALL,XAB$C_ALLLEN,XBC$C_CREALL1
00'24 12 0003 214 .BYTE XAB$C_DAT,XAB$C_DATLEN_V2,XBC$C_OPNDAT
00'2C 1D 0006 215 .BYTE XAB$C_FHC,XAB$C_FHCLEN,XBC$C_OPN_FHC
00'10 13 0009 216 .BYTE XAB$C_PRO,XAB$C_PROLEN_V3,XBC$C_CREPRO
00'14 1E 000C 217 .BYTE XAB$C_RDT,XAB$C_RDTLEN,XBC$C_OPNRDT
00'3C 22 000F 218 .BYTE XAB$C_JNL,XAB$C_JNLLEN,XBC$C_CREJNL
      00 0012 219 .BYTE 0
      0013 220
      0013 221 ;
      0013 222 ; arguments for "all" xab processing to return alq from actual allocated size
      0013 223 ;
      0013 224
      0013 225 CRE_XAB_ARGS2:
00'20 14 0013 226 .BYTE XAB$C_ALL,XAB$C_ALLLEN,XBC$C_CREALL2
00'10 13 0016 227 .BYTE XAB$C_PRO,XAB$C_PROLEN_V3,XBC$C_CREPRO1
      00 0019 228 .BYTE 0
      001A 229
      001A 230 ;
      001A 231 ; Table to associate action routines for attributes when journaling $CREATE.
      001A 232 ;
      001A 233
      001A 234 ;
      001A 235 ; First, offsets for case branch.
      001A 236 ;
00000000 001A 237 TAB$C_USER_CHAR = 0
0000C001 001A 238 TAB$C_FILE_PRO = 1
00000002 001A 239 TAB$C_FILE_UIC = 2
00000003 001A 240 TAB$C_RECORD_ATTR = 3
00000004 001A 241 TAB$C_EXPIRE = 4
      001A 242
      001A 243
      001A 244 TABLE:
00000000 00000003 001A 245 .LONG ATR$C_UCHAR, TAB$C_USER_CHAR ; user characteristics
00000001 00000016 0022 246 .LONG ATR$C_FPRO, TAB$C_FILE_PRO ; file's protection
00000002 00000015 002A 247 .LONG ATR$C_UIC, TAB$C_FILE_UIC ; file's UIC
00000003 00000004 0032 248 .LONG ATR$C_RECATTR, TAB$C_RECORD_ATTR ; record attributes
00000004 00000013 003A 249 .LONG ATR$C_EXPDATE, TAB$C_EXPIRE ; expiration date
00000000 00000000 0042 250 .LONG 0, 0 ; end of table
      004A 251

```



```

004A 253      .SBTTL RM$CREATECOM - PERFORM CREATE FILE FUNCTION
004A 254
004A 255      :++
004A 256      : RM$CREATECOM
004A 257
004A 258      : RM$CREATECOM -
004A 259
004A 260      : this routine sets up the fib from the various user options, directory id and
004A 261      : allocation information, builds the attribute list to write the user record
004A 262      : and other attributes, builds the qio parameter list on the stack
004A 263      : issues the qio to the acp to perform the create (or access if 'cif' bit set)
004A 264      : initializes the hbk field of the ifab, and finally calls rm$fillnam to retur
004A 265      : the resultant name string.
004A 266
004A 267      : this routine also calls subroutines to journal the $CREATE, if
004A 268      : journaling is taking place. one subroutine collects all the information
004A 269      : necessary to journal, another actually writes the journal entries
004A 270      : to the open journals.
004A 271
004A 272      : Calling sequence:
004A 273
004A 274      :     bsbw      rm$createcom
004A 275
004A 276      : Input Parameters:
004A 277
004A 278      :     r11      impure area addr
004A 279      :     r10      fwa address
004A 280      :     r9       ifab address
004A 281      :     r8       fab address
004A 282
004A 283      : Implicit Inputs:
004A 284
004A 285      :     the contents of the parameter blocks listed
004A 286      :     above, especially:
004A 287
004A 288      :     ifb$l_prim_dev
004A 289      :     fwa$q_dir
004A 290      :     fwa$q_name
004A 291      :     fwa$l_atrladr
004A 292      :     ifb$l_chnl
004A 293      :     ifb$b_fac
004A 294      :     ifb$l_rfmorg thru ifb$sc_fhaend
004A 295      :     fab$l_fop
004A 296      :     fab$l_alq
004A 297      :     fab$l_xab
004A 298      :     fab$l_nam
004A 299
004A 300
004A 301      : Output Parameters:
004A 302
004A 303      :     r0       status code
004A 304      :     r6       fib addr
004A 305      :     r1-r5,r7,ap destroyed
004A 306
004A 307      : Implicit Outputs:
004A 308
004A 309      :     ifb$u_accessed set

```

```

004A 310 :      nam$l_rsl set to length of resultant string, if any
004A 311 :      ifb$l_ios
004A 312 :      ifb$l_hbk
004A 313 :      fab$l_stv set to system error code on failure
004A 314 :
004A 315 :      Completion Codes:
004A 316 :
004A 317 :      standard rms, including suc, rer, wer,
004A 318 :      flk, prv, dnf, ful, and cre.
004A 319 :
004A 320 :      Side Effects:
004A 321 :
004A 322 :      may have switched to running at ast level.
004A 323 :      all user structures except fab and nam
004A 324 :      must be reprobbed.
004A 325 :--
004A 326 :

```



```
004A 328 RMSCREATECOM::
06 03 E0 004A 329 BBS #DEV$V DIR,-
06 69 004C 330 IFB$L_PRIM_DEV(R9),10$ ; branch if files-oriented
017A 31 004E 331 RMSSUC ; show success
0051 332 BRW FILNAM ; fill in nam block
0054 333
0054 334 ; unless 'tmp' or 'tmd' set in fop, get the directory id
0054 335
0054 336
0054 337
04 18 D3 0054 338 10$: BITL #FAB$M_TMP!FAB$M_TMD,-
04 A8 0056 339 FAB$L_FOP(R8) ; tmp or tmd set?
0D 12 0058 340 BNEQ SETTMP ; branch if yes
FFA3' 30 005A 341 BSBW RMS$SETDID ; get directory id
0B 50 E8 005D 342 BLBS RO,SETUP ; continue if ok
05 0060 343 CREXIT: RSB ; get out on error
0061 344
0061 345 ; handle bad alq value
0061 346
0061 347
0061 348
0061 349 ERRALQ:
0061 350 RMSERR ALQ
05 0066 351 RSB
0067 352
0067 353 ; set tmp bit to flag this as a temporary file
0067 354
0067 355
0067 356
0067 357 SETTMP: SSB #IFB$V_TMP,(R9) ; flag temporary file
006B 358
006B 359 ; call rm$creacc_set1 to do common fib and attribute list set up
006B 360
006B 361
006B 362
FF92' 30 006B 363 SETUP: BSBW RMS$CREACC_SET1
EF 50 E9 006E 364 BLBC RO,CREXIT
44 AA B4 0071 365 CLRW FWAS$_UCHAR(R10) ; initialize user characteristics
0074 366
0074 367 ;
0074 368 ; r6 now points to fib
0074 369 ; r5 has address of where to build attribute list entries
0074 370
0074 371 ;!!!!!!
0074 372 ;\must include code to handle magtape label xabs.\
0074 373 ;!!!!!!
0074 374
0074 375 ; handle xabs
0074 376
0074 377
5C 89 AF 9E 0074 378 MOVAB CRE_XAB_ARGS,AP ; set xab scan args addr
FF85' 30 0078 379 BSBW RMS$XAB_SCAN ; process xabs
E2 50 E9 007B 380 BLBC RO,CREXIT ; get out on error
05 E0 007E 381 10$: BBS #DEV$V_SQD,-
36 69 0080 382 IFB$L_PRIM_DEV(R9),50$ ; branch if magtape
0082 383
0082 384 ;
```

```
0082 385 ; if disk, process initial allocation request
0082 386 ;
0082 387 ;
10 A8 D0 0082 388 20$: MOVL FAB$L_ALQ(R8),-
18 A6 13 0085 389 FIB$L_EXSZ(R6) ; set alloc size
2F 19 0087 390 BEQL 50$ ; branch if none
D6 0089 391 BLSS ERRALQ ; branch if bad
06 68 31 E1 008B 392 BBC #FAB$V_UFO+FOP,(R8),30$ ; branch if not ufo
10 A8 01 C1 008F 393 ADDL3 #1,FAB$L_ALQ(R8),-
74 A9 0093 394 IFB$L_EBK(R9) ; set eof blk from alq
0095 395 30$: SSB #FIB$V_EXTEND,-
0095 396 FIB$W_EXCTL(R6) ; enable extend
1A 54 00' E0 009A 397 BBS S^#XBC$C_CREALL1,R4,50$ ; branch if alq xab seen
09 68 35 E1 009E 398 BBC #FAB$V_CBT+FOP,(R8),40$ ; branch if cbt bit off
20 88 00A2 399 BISB2 #1@FCH$V_CONTIGB,-
44 AA 00A4 400 FWASW_UCHAR(R10) ; give file cbt attribute
01 E3 00A6 401 BBBS #FIB$V_ALCONB,-
0D 16 A6 00AB 402 FIB$W_EXCTL(R6),50$ ; ask primitive for best try
09 68 34 E1 00AB 404 40$: BBC #FAB$V_CTG+FOP,(R8),50$ ; branch if contig bit off
05 88 00AF 405 BISB2 #FIB$M_ALCON!FIB$M_FILCON,-
16 A6 00B1 406 FIB$W_EXCTL(R6) ; ask acp for ctg extend
80 8F 88 00B3 407 BISB2 #1@FCH$V_CONTIG,-
44 AA 00B6 408 FWASW_UCHAR(R10) ; give file ctg attribute
00B8 409
00B8 410 ;
00B8 411 ; swap words of ebk to be files-11 compatible and insert org into rfm/org
00B8 412 ; so that it gets written with the file attributes.
00B8 413 ; (note: hbk is zero)
00B8 414 ;
00B8 415 ;
50 A9 58 A9 74 A9 10 9C 00B8 416 50$: ROTL #16,IFB$L_EBK(R9),IFB$L_EBK_DISK(R9)
04 04 23 A9 F0 00BE 417 INSV IFB$B_ORG$CASE(R9),#IFB$V_ORG,#IFB$S_ORG,IFB$B_RFMORG(R9)
00C5 418
00C5 419 ;
00C5 420 ; If the alignment option is RFI with a file id of zero, it means create
00C5 421 ; the file next to itself. Since the ACP is going to try to look for a
00C5 422 ; file that does not exist yet, make the option ANY, which is really what
00C5 423 ; it should be mean on a create.
00C5 424 ;
00C5 425 ;
04 21 A6 91 00C5 426 CMPB FIB$B_ALALIGN(R6),#FIB$C_RFI ; RFI option?
0E 12 00C9 427 BNEQ 60$ ; branch if not
22 A6 D5 00CB 428 TSTL FIB$W_LOC_FID(R6) ; zero file id
09 12 00CE 429 BNEQ 60$ ; branch if not
26 A6 D5 00D0 430 TSTL FIB$W_LOC_RVN(R6)
04 12 00D3 431 BNEQ 60$
21 A6 00 90 00D5 432 MOVB #XAB$C_ANY,FIB$B_ALALIGN(R6) ; make it ANY
00D9 433
00D9 434 ;
00D9 435 ; Enforce RU bit settings, specifically ONLY_RU
00D9 436 ;
00D9 437 60$:
00A0 C9 51 DD 00D9 438 PUSHL R1 ; Save R1 first
00A0 C9 20 8A 00DB 439 BICB2 #IFB$M_NEVER_RU,IFB$B_JNLFLG(R9) ; Ignoring NEVER_RU, is
03 93 00E0 440 BITB #IFB$M_RU!IFB$M_ONLY_RU,IFB$B_JNLFLG(R9) ; any RU-bit set?
1E 13 00E5 441 BEQL 62$ ; If not, go on with stuff
```



```

51 00000000'9F D0 00E7 442 MOVL @#CTL$GL_RUF,R1 ; RUF loaded?
      05 13 00EE 443 BEQL 61$ ; No RUF, check for ONLY_RU
10 11 A1 01 E0 00F0 444 BBS #RUCB$V_ACTIVE,RUCB$B_CTRL(R1),62$ ; In RU? Then we're cool
      01 93 00F5 445 61$: BITB #IFB$M_ONLY_RU,- ; If ONLY_RU clear (RU
      00A0 C9 00F7 446 ; IFB$B_JNLFLG(R9) ; must be set), and not
      09 13 00FA 447 BEQL 62$ ; in RU then that's ok
      51 8ED0 0101 448 RMSERR NRU ; However, if ONLY_RU set and not in RU: error
      05 0104 449 POPL R1 ; Realign stack
      0105 450 RSE
      0105 451
      0105 452
      0105 453 ;
      0105 454 ; process the mxv, sup, and cif options
      0105 455 ;
      0105 456 ; for mxv & sup, need merely copy to fib
      0105 457 ;
      51 8ED0 0105 458 62$: POPL R1 ; Pop R1
      0108 459 ASSUME FAB$V_SUP EQ FAB$V_MXV+1
      0108 460 ASSUME FIB$V_SUPERSEDE EQ FIB$V_NEWVER+1
14 50 68 02 21 EF 0108 461 NAMCTL: EXTZV #FAB$V_MXV+FOP,#2,(R8),R0
      02 09 50 F0 010D 462 INSV R0,#FIB$V_NEWVER,#2,FIB$V_NMCTL(R6)
      03 68 39 E1 0113 463 BBC #FAB$V_CIF+FOP,(R8),SET3
      0116 31 0117 464 BRW DOCIF
      011A 465
      011A 466 ;
      011A 467 ; call rm$creacc_set3 to finish building the qio parameters for create
      011A 468 ;
      011A 469 ;
      FEE3' 30 011A 470
      471 SET3: BSBW RM$CREACC_SET3

```



```

011D 473
011D 474
011D 475 : set i/o function code and do the create
011D 476
011D 477
F3 8F 9A 011D 478 MOVZBL #IOS_CREATE!IOSM_CREATE!IOSM_ACCESS,-
04 68 50 0120 479 R0 ; set qio function code
24 E1 0121 480 BBC #FAB$V_TMD+FOP,(R8),10$ ; branch unless tmd set
FED4' 30 0125 481 SSB S^IOSV_DELETE,R0 ; mark file for delete
03 50 E8 0129 482 10$: BSBW RMS$FCP$NC ; do the create
0133 31 012C 483 BLBS R0,GETJNL ; branch if ok
012F 484 BRW ERRCREATE ; branch on error
0132 485
0132 486
0132 487
0132 488 : Journal the $CREATE if journaling. Then retrieve journal names from
0132 489 : XAB and mark the file with them.
0132 490
0132 491
0132 492 GETJNL:
00A0 C9 50 DD 0132 493 PUSHL R0 ; save status code
00A0 C9 20 CA 0134 494 BICL2 #IFB$M_NEVER_RU,IFB$B_JNLFLG(R9) ; don't care about NEVER_RU
5D 69 05 E0 0139 495 TSTB IFB$B_JNLFLG(R9) ; journaling bits seen?
013D 496 BEQL SETHBR_BR ; branch if not
013F 497 BBS #DEV$V_SQD,IFB$L_PRIM_DEV(R9),ERRJOP ; no disk, no journaling
0143 498
0143 499 : Collect journaling information from current attribute list.
0143 500
0143 501
00000000'EF 30 BB 0143 501 PUSHR #^M<R4,R5> ; save work regs
16 0145 502 JSB RMS$ALLOC_RJB_BDB ; get an RJB
30 BA 0148 503 POPR #^M<R4,R5> ; restore work regs
55 50 E9 014D 504 BLBC R0,BR_AID ; out on error
011A 30 0150 505 BSBW RMS$INT_CRE_RJR ; set up the $CREATE RJR
0153 506
0153 507
0153 508 : Mark the new file for journaling as specified in the XAB.
0153 509
55 58 AA D0 0153 510 MOVL FWASL_ATR_WORK(R10),R5 ; set up new attr list.
00000000'EF 16 0157 511 JSB RMS$GETJNL ; get journal names for create
45 50 E9 015D 512 BLBC R0,BR_AID ; get out on error
0160 513
0160 514 : Assign channels to the appropriate journals.
0160 515
0160 516
00000000'EF 30 BB 0160 516 PUSHR #^M<R4,R5> ; save XAB flags, attr address
16 0162 517 JSB RMS$ASSJNL ; assign journal channels and init
0168 518 : journal data structures
30 BA 0168 519 POPR #^M<R4,R5> ; restore XAB flags, attr address
38 50 E9 016A 520 BLBC R0,BR_AID ; get out on error
016D 521
016D 522 : Journal the $CREATE to the appropriate journals.
016D 523
016D 524
01B2 30 016D 524 BSBW RMS$JNL_CREATE ; write the RJR to the journals
32 50 E9 0170 525 BLBC R0,BR_AID ; get out on error
0173 526
7E 16 A6 3C 0173 527 MOVZWL FIB$W_EXCTL(R6),-(SP) ; save EXCTL
7E 18 A6 7D 0177 528 MOVQ FIB$L_EXSZ(R6),-(SP) ; save EXSZ, EXVBN
16 A6 B4 017B 529 CLRW FIB$W_EXCTL(R6) ; reset EXCTL to zero

```



```

18 A6 7C 017E 530 ASSUME FIB$L_EXVBN EQ <FIB$L_EXSZ+4>
65 D4 017E 531 CLRQ FIB$L_EXSZ(R6) ; also zero EXSZ, EXVBN
00 DD 0181 532 CLRL (R5) ; indicate end of attr list
58 AA DD 0183 533 PUSHL #0 ; set up QIO P6
FE75' 30 0185 534 PUSHL FWASL_ATR_WORK(R10) ; QIO P5 attr list
50 FE6F' 9A 0188 535 BSBW RMS$FCP_P4-P2 ; QIO P4,P3,P2
FE6F' 30 018B 536 MOVZBL #IOS_MODIFY,R0 ; set modify function
0191 537 BSBW RMS$FCPFNC ; do the modify
18 A6 8E 7D 0191 538 MOVQ (SP)+,FIB$L_EXSZ(R6) ; restore EXSZ, EXVBN
16 A6 6E B0 0195 540 MOVW (SP),FIB$W_EXCTL(R6) ; restore EXCTL
8E D5 0199 541 TSTL (SP)+ ; pop EXCTL
09 50 E9 019B 542 BLBC R0,ERRMOD ; get out on error
019E 543 SETHBK_BR: SETHBK ; continue with create
11 11 019E 544 BRB
01A0 545
43 11 01A0 546 ERRJOP: RMSERR JOP
01A5 547 BR_AID: BRB XITPOP
01A7 548
FE51' 30 01A7 549 ERRMOD: RMSERR CRE,R1
39 11 01AC 550 BSBW RMS$MAPERR
01AF 551 BRB XITPOP
01B1 552
18 A6 D0 01B1 553 SETHBK: MOVL FIB$L_EXSZ(R6),-
70 A9 01B4 555 IFB$L_HBK(R9) ; set hi block
5C FE59 CF 9E 01B6 556 MOVAB CRE_XAB_ARGS2,AP ; set xab arglist addr
FE42' 30 01BB 557 BSBW RMS$XAB_SCAN ; go set alq in xaball
29 50 E9 01BE 558 BLBC R0,XITPOP ; get out on error
01C1 559
FE3C' 30 01C1 560 SETEBK: BSBW RMS$SETEBK ; go set ebk, accessed
26 50 E9 01C4 561 BLBC R0,DELSHR ; delete on error
01 01C7 562 POPR #^M<R0> ; restore status code
01C9 563 ASSUME IFB$V_RFM EQ 0
01C9 564 ASSUME IFB$S_RFM EQ 4
50 A9 F0 8F 8A 01C9 565 BICB2 #^XF0,IFB$B_RFMORG(R9) ; leave only rfm in rfmorg
01CE 566
50 DD 01CE 567 FILNAM: PUSHL R0 ; save success code
FE2D' 30 01D0 568 BSBW RMS$FILLNAM ; return resultant string
14 50 E9 01D3 569 BLBC R0,XITPOP ; branch on error
57 D5 01D6 570 TSTL R7 ; is there a nam blk?
0D 13 01D8 571 BEQL 10$ ; branch if none
01DA 572
01DA 573 ;
01DA 574 ; set the lower and higher flags in the nam block
01DA 575 ;
01DA 576
01DA 577 ASSUME FIB$V_HIGHVER EQ FIB$V_LOWVER+1
01DA 578 ASSUME NAM$V_HIGHVER EQ NAM$V_LOWVER+1
02 0E EF 01DA 579 EXTZV #FIB$V_LOWVER,#2,-
0208 CA 01DD 580 FIB$W_NMCTL+FWASL_FIBBUF(R10),-
51 01E0 581 R1 ; get version bits
02 0E 51 F0 01E1 582 INSV R1,NAM$V_LOWVER,#2,-
34 A7 01E5 583 NAM$L_FNB(R7) ; and set in nam blk
01 01E7 584 10$: POPR #^M<R0> ; restore success code
05 01E9 585 RSB
01EA 586

```



```

02  BA 01EA 587 XITPOP: POPR    #^M<R1>          ; remove success code
    05 01EC 588 EXIT:  RSB          ; and return with error
    01ED 589
    01ED 590
    01ED 591
    01ED 592
    01ED 593
    01ED 594
    50  DD 01ED 595 DELSHR: PUSHL    R0          ; save status code
52  40 8F 9A 01EF 596 MOVZBL    #FIB$C_LENGTH,R2      ; get length of fib
    FE06' 30 01F3 597 CSB        #IMP$V_TEMP1,(R11)    ; clear s0 flag
    04 50 E8 01FA 598 BSBW      RM$GETSPC1          ; go get fib
    02  BA 01FD 600 BLBS        R0,20$              ; branch if ok
    E9  11 01FF 601 POPR        #^M<R1>              ; clean up stack
    51  DD 0201 602 20$: PUSHL    R1                ; leave
7E  40 8F 9A 0203 603 MOVZBL    #FIB$C_LENGTH,-(SP)    ; push fib address
11 69 35 E0 0207 604 BBS        #IFB$V_TMP,(R9),DEL    ; push length of fib
    020B 605 SSB        #FIB$V_FINDFID,-            ; branch if temp file
    020B 606 FIB$W_NMCTL(R1) ; set findfid bit
    01FE CA B0 0210 607 MOVW     FIB$W_DID+FWAST_FIBBUF(R10),- ; set did
    0A A1 0214 608 FIB$W_DID(R1) ; set did
    0200 CA D0 0216 609 MOVL     FIB$W_DID_SEQ+FWAST_FIBBUF(R10),- ; set did sequence and rvn
    0C A1 021A 610 FIB$W_DID_SEQ(R1) ; set did sequence and rvn
    0135 8F 3C 021C 611 DEL: MOVZWL #<IOS_DELETE!IOSM_DELETE>,- ; set i/o func code
    50 0220 612 R0          ; set p6 = p5 = 0 for $qio
    7 7C 0221 613 CLRQ        -(SP)                ; go do the delete
    FDD' 30 0223 614 BSBW      RM$FCPFNC_P4          ; get fib addr, length
    14 BA 0226 615 POPR        #^M<R2,R45>          ; return the space
    FDD5' 30 0228 616 BSBW      RM$RETSPC1          ; restore registers
    03 BA 022B 617 POPR        #^M<R0,R1>          ; close cleanup
    FDD0' 31 022D 618 BRW      RM$CLSCU

```

this code will only be executed if we return an error status from
rm\$open_shared which is called in the subroutine rm\$setebk.


```

0230 620
0230 621 :++
0230 622 :
0230 623 : 'cif' bit is set indicating open file if it exists,
0230 624 : otherwise create - do remaining setup & access
0230 625 :
0230 626 :--
0230 627 :
06 69 05 E0 0230 628 DOCIF: BBS #DEV$V SQD,IFB$$_PRIM_DEV(R9),10$ ; branch if not disk
00000000'EF 16 0234 629 JSB RMSRTVJNL ; set up for read of journal names
FDC3' 30 023A 630 10$: BSBW RM$CREACC SET2 ; finish param setup
F2 8F 9A 023D 631 MOVZBL #IOS_ACCESS!IOSM_CREATE!IOSM_ACCESS,-
50 0240 632 RO ; set qio function code
04 68 24 E1 0241 633 BBC #FAB$V TMD+FOP,(R8),20$ ; if tmd not set
0245 634 SSB S^#IOS$ DELETE,RO ; mark file for delete
FDB4' 30 0249 635 20$: BSBW RM$FCPFNC ; do access/create
16 50 E9 024C 636 BLBC RO,ERRCREATE ; branch on error
01 50 B1 024F 637 CMPW RO,S^#SS$_NORMAL ; was file created?
03 13 0252 638 BEQL 60$ ; branch if not
0254 639
FEDB 31 0254 640 BRW GETJNL ; re-join create
0257 641
0257 642
0257 643 :
0257 644 : 'cif' and file already existed - switch to 'open'
0257 645 :
0257 646 :
02 91 0257 647 60$: CMPB #IFB$C_IDX,-
23 A9 0259 648 IFB$B_ORGCASE(R9) ; indexed, rm$create3b
05 13 025B 649 BEQL 70$ ;
6E 0000'CF 9E 025D 650 MOVAB W^RM$OPEN_CIF,(SP) ; change return pc
FD9B' 31 0262 651 70$: BRW RM$SETHBK ; & go finish up open
0265 652
0265 653 :
0265 654 : process error on create
0265 655 :
0265 656 :
0265 657 ERRCREATE:
FD93' 31 0265 658 RMSERR CRE,R1 ; default error code
026A 659 BRW RM$MAPERR ; go map the error
026D 660

```

```

026D 662 .SUBTITLE RMSINI_CRE_RJR
026D 663 :++
026D 664 : RMSINI_CRE_RJR
026D 665 :
026D 666 : This routine is used to fill in the necessary information to re-do
026D 667 : a $CREATE operation.
026D 668 :
026D 669 : Input Parameters:
026D 670 :
026D 671 : r9 - IFAB
026D 672 : r10 - FWA
026D 673 :
026D 674 : Implicit Inputs:
026D 675 :
026D 676 : JNLBDB - for RJR
026D 677 :
026D 678 : Output Parameters:
026D 679 :
026D 680 : r0 - status
026D 681 :
026D 682 : Implicit Outputs:
026D 683 :
026D 684 : RJR filled with info required to re-do $CREATE.
026D 685 :
026D 686 : Side Effects:
026D 687 :
026D 688 : None.
026D 689 :
026D 690 :--
026D 691 :
026D 692 RMSINI_CRE_RJR:
026D 693 :
00BC 8F BB 026D 694 PUSH R2,R3,R4,R5,R7 ; save work registers
50 D4 0271 695 CLRL R0 ; anticipate the worst
0273 696 :
0273 697 : Get RJR address.
0273 698 :
54 30 A9 D0 0273 699 MOVL IFB$L_JNLBDB(R9),R4 ; get address of journaling BDB
67 13 0277 700 BEQL 50$ ; get out if none
57 18 A4 D0 0279 701 MOVL BDB$L_ADDR(R4),R7 ; get RJR address
61 13 027D 702 BEQL 50$ ; get out if none
027F 703 :
027F 704 :
027F 705 : Handle file attributes.
027F 706 :
55 58 AA D0 027F 707 MOVL FWA$L_ATTR_WORK(R10),R5 ; get address of attribute list
0283 708 10$: TSTW ATTR$W_TYPE(R5) ; is an attribute present?
02  A5 B5 0283 709 BEQL 40$ ; if eql, no - all done
2D 13 0286 710 MOVAL TABLE,R4 ; get address of table of attributes
54 FD8E CF DE 0288 711 ; to journal (and their action routines)
028D 712 :
028D 713 15$: CMPW JNL$W_ATTR_TYPE(R4),- ; is this table entry identical to this
02  A5 B1 028D 714 ATR$W_TYPE(R5) ; entry in the attribute list?
16 12 028F 715 BNEQ 20$ ; if NEQ no, not equal
0293 716 :
000002B0'EF 9F 0293 717 PUSHAB 30$ ; push return address from CASE
0293 718 :

```



```
0299 719 CASE TYPE = B,-
0299 720 SRC = JNL$ ACTION RTN(R4),-
0299 721 DISPLIST = <USER_CHAR,FILE_PRO,FILE_UIC,RECORD_ATTR,EXPIRE>
02A8 722
05 02A8 723 RSB ; to 30$ if bad case offset
54 08 C0 02A9 724 20$: ADDL2 #JNL$C_ENT_SIZE,R4 ; point to next entry in table
64 64 B5 02AC 725 TSTW JNL$W_ATTR_TYPE(R4) ; is there another entry in table?
DD 12 02AE 726 BNEQ 15$ ; yes, go compare with attribute
55 08 C0 02B0 728 30$: ADDL2 #ATTR$S_ENT_SIZE,R5 ; point to next attribute
CE 11 02B0 729 BRB 10$ ; go process it
02B3 730
02B5 731 ; all done with attributes
02B5 732 40$:
02B5 733
02B5 734 ; Copy FIB.
02B5 735 ;
02B5 736 ;
64 A7 14 BA 10 AA 28 02B5 737 MOV C3 FWA$Q_FIB(R10),@FWA$Q_FIB+4(R10),RJR$T_C_FIB(R7)
02B8 738
02B8 739 ; Copy Filename.
02B8 740 ;
02B8 741 ;
53 00C4 C7 DE 02B8 742 MOVAL RJR$T_FILENAME(R7),R3 ; get name buff addr
02C1 743
02C1 744 ASSUME RJR$S_FILENAME EQ 256
02C1 745
02C1 746 ;
02C1 747 ; Set buffer size to 255 because the GETFILNAM code builds a NAM block, etc...
02C1 748 ; and can only cope with a size that fits in a byte.
02C1 749 ;
54 00FF 8F 3C 02C1 750 MOVZWL #<RJR$S_FILENAME-1>,R4 ; set size of buffer
00000000'EF 16 02C6 751 JSB RMS$GETFILNAM ; go get file name
58 A7 54 90 02CC 752 MOV B R4,RJR$B_FNS(R7) ; put length in entry
02D0 753
02D0 754 ;
02D0 755 ; Fill in the rest of the journal record.
02D0 756 ;
04 A7 23 A9 90 02D0 757 MOV B IFB$B_ORGCASE(R9),RJR$B_ORG(R7) ; file organization
03 A7 01 90 02D5 758 MOV B #RJR$C_FILENAME,RJR$B_ENTRY_TYPE(R7) ; filename
05 A7 04 90 02D9 759 MOV B #RJR$_CREATE,RJR$B_OPER(R7) ; RMS operation
02DD 760
50 01 D0 02DD 761 MOVL #1,R0 ; indicate success
00BC 8F BA 02E0 762 50$: POPR #^M<R2,R3,R4,R5,R7> ; restore work registers
05 02E4 763 RSB ; to caller
```



```
02E5 765 .SUBTITLE Routines to journal CREATE attributes.
02E5 766 :++
02E5 767 :
02E5 768 : Action routines for journaling $CREATE attributes.
02E5 769 :
02E5 770 : Inputs:
02E5 771 :
02E5 772 : R5 points to attribute.
02E5 773 : R7 points to RJR.
02E5 774 :
02E5 775 : Outputs:
02E5 776 :
02E5 777 : RJR CREATE attributes filled in, MASK longword also filled in.
02E5 778 :
02E5 779 : Side Effects:
02E5 780 :
02E5 781 : R4 and R5 must be preserved.
02E5 782 :
02E5 783 :--
02E5 784 :
02E5 785 :++
02E5 786 : User Characteristics
02E5 787 :--
02E5 788 :
02E5 789 USER_CHAR: ; user characteristics
02E5 790 SSB #RJR$V_ATR UCHAR,RJR$L_ATR FLAGS(R7) ; indicate UCHAR
4C A7 04 B5 D0 02EA 791 MOVL @ATR$L_ADDR(R5),RJR$L_UCHAR(R7) ; copy characteristics
05 02EF 792 RSB ; to main routine
02F0 793 :
02F0 794 :++
02F0 795 : File Protection
02F0 796 :--
02F0 797 :
02F0 798 FILE_PRO: ; file protection
02F0 799 SSB #RJR$V_ATR PROT,RJR$L_ATR FLAGS(R7) ; indicate PROT
44 A7 04 B5 D0 02F5 800 MOVL @ATR$L_ADDR(R5),RJR$L_PROT(R7) ; copy protection
05 02FA 801 RSB ; to main routine
02FB 802 :
02FB 803 :++
02FB 804 : File's UIC
02FB 805 :--
02FB 806 :
02FB 807 FILE_UIC: ; file's UIC
02FB 808 SSB #RJR$V_ATR UIC,RJR$L_ATR FLAGS(R7) ; indicate UIC
40 A7 04 B5 D0 0300 809 MOVL @ATR$L_ADDR(R5),RJR$L_UIC(R7) ; copy UIC
05 0305 810 RSB ; to main routine
0306 811 :
0306 812 :++
0306 813 : Record Attributes
0306 814 :--
0306 815 :
0306 816 RECORD_ATTR: ; record attribute block
0306 817 PUSH R4,R5 ; save pointers
0308 818 SSB #RJR$V_ATR REC,RJR$L_ATR FLAGS(R7) ; indicate REC
030D 819 MOV C3 #RJR$C_REC_ATTRLEN,- ; copy the record attributes
04 B5 030F 820 MOVL @ATR$L_ADDR(R5),- ; ...
00A4 C7 0311 821 RJR$L_REC_ATTR(R7) ; ...
```



```

30  BA 0314 822      POPR    #^M<R4,R5>      ; restore pointers
    05 0316 823      RSB      ; to main routine
        0317 824
        0317 825 :++
        0317 826 : Expiration Date
        0317 827 :--
        0317 828
        0317 829 EXPIRE:
50 A7 04 B5 7D 0317 830      SSB      ; expiration date
    05 031C 831      MOVQ    #RJR$V_ATR EXPIRE,RJR$L_ATR_FLAGS(R7) ; indicate EXPIRE
        0321 832      RSB      @ATR$L_ADDR(R5),RJR$Q_EXPIRE(R7) ; copy expiration date
        0322 833      ; to main routine

```

```
0322 835 .SUBTITLE RMSJNL_CREATE
0322 836 :++
0322 837 : RMSJNL_CREATE
0322 838 :
0322 839 : This routine writes create journal entries for AI, BI, or RU journal.
0322 840 : AT creates are done later.
0322 841 :
0322 842 : Input Parameters:
0322 843 :
0322 844 :     r9      - IFAB
0322 845 :
0322 846 : Implicit Inputs:
0322 847 :
0322 848 :     JNLBDB - to describe journal buffer used for $CREATE
0322 849 :     RJB    - to see if journaling is turned on, input to RMSWRITEJNL
0322 850 :
0322 851 : Output Parameters:
0322 852 :
0322 853 :     r0      - status of operation
0322 854 :
0322 855 : Implicit Outputs:
0322 856 :
0322 857 :     $CREATE journal entries are written.
0322 858 :
0322 859 : Side Effects:
0322 860 :
0322 861 :     None.
0322 862 :
0322 863 :--
0322 864 :
0322 865 RMSJNL_CREATE:
0322 866
0322 867     PUSH    #M<R4,R5,R6>          ; save work registers
0322 868     MOVL    #1,R6                 ; anticipate success
0322 869     MOVL    IFB$L_JNLBDB(R9),R4   ; get address of BDB
0322 870     MOVL    IFB$L_RJB(R9),R5     ; and get address of RJB
0322 871
0322 872     MOVW    #RJR$C_FILNAMLEN,BDB$W_NUMB(R4) ; size of buffer to write
0322 873     MOVQ    R3,-(SP)              ; init input to RMSWRITEJNL with
0322 874                                     ; longword BDB address and longword
0322 875                                     ; to be overwritten by journal type
0322 876     BBC     #RJB$V_BI,RJB$W_FLAGS(R5),10$ ; branch if no BI
0322 877     MOVZBL  #CJF$_BI,(SP)         ; type of journal
0322 878     JSB     RMSWRTJNL_OBJ         ; write jnl entry as OBJECT_ID
0322 879     BLBS    R0,10$               ; skip on success
0322 880     MOVL    R0,R6                ; save error code
0322 881 10$:
0322 882     BBC     #RJB$V_AI,RJB$W_FLAGS(R5),20$ ; branch if no AI
0322 883     MOVZBL  #CJF$_AI,(SP)         ; type of journal
0322 884     JSB     RMSWRTJNL_OBJ         ; write jnl entry as OBJECT_ID
0322 885     BLBS    R0,20$               ; skip on success
0322 886     MOVL    R0,R6                ; save error code
0322 887 20$:
0322 888     BBC     #RJB$V_RU,RJB$W_FLAGS(R5),30$ ; branch if no RU
0322 889     MOVL    @#CTL$GL_RUF,R1       ; don't write entry if no active RU
0322 890     BEQL    30$                  ; no RUCB, no RU
0322 891     BBC     #RUCB$V_ACTIVE,RUCB$B_CTRL(R1),30$ ; no active RU
```

0070 8F BB	0322 867	PUSHR	#M<R4,R5,R6>	; save work registers
56 01 DO	0326 868	MOVL	#1,R6	; anticipate success
54 30 A9 DO	0329 869	MOVL	IFB\$L_JNLBDB(R9),R4	; get address of BDB
55 00A4 C9 DO	032D 870	MOVL	IFB\$L_RJB(R9),R5	; and get address of RJB
	0332 871			
14 A4 01C4 8F B0	0332 872	MOVW	#RJR\$C_FILNAMLEN,BDB\$W_NUMB(R4)	; size of buffer to write
7E 53 7D	0338 873	MOVQ	R3,-(SP)	; init input to RMSWRITEJNL with
	033B 874			; longword BDB address and longword
	033B 875			; to be overwritten by journal type
OF OA A5 01 E1	033B 876	BBC	#RJB\$V_BI,RJB\$W_FLAGS(R5),10\$; branch if no BI
6E 02 9A	0340 877	MOVZBL	#CJF\$_BI,(SP)	; type of journal
00000000 EF 16	0343 878	JSB	RMSWRTJNL_OBJ	; write jnl entry as OBJECT_ID
03 50 E8	0349 879	BLBS	R0,10\$; skip on success
56 50 DO	034C 880	MOVL	R0,R6	; save error code
	034F 881			
OF OA A5 02 E1	034F 882	BBC	#RJB\$V_AI,RJB\$W_FLAGS(R5),20\$; branch if no AI
6E 03 9A	0354 883	MOVZBL	#CJF\$_AI,(SP)	; type of journal
00000000 EF 16	0357 884	JSB	RMSWRTJNL_OBJ	; write jnl entry as OBJECT_ID
03 50 E8	035D 885	BLBS	R0,20\$; skip on success
56 50 DO	0360 886	MOVL	R0,R6	; save error code
	0363 887			
1A OA A5 00 E1	0363 888	BBC	#RJB\$V_RU,RJB\$W_FLAGS(R5),30\$; branch if no RU
51 00000000 9F DO	0368 889	MOVL	@#CTL\$GL_RUF,R1	; don't write entry if no active RU
11 13	036F 890	BEQL	30\$; no RUCB, no RU
OC 11 A1 01 E1	0371 891	BBC	#RUCB\$V_ACTIVE,RUCB\$B_CTRL(R1),30\$; no active RU

6E	01	9A	0376	892	
00000000	EF	16	0379	893	
56	50	D0	037F	894	
			0382	895	30\$:
50	56	D0	0382	896	
5E	08	C0	0385	897	
0070	8F	BA	0388	898	
		05	038C	899	
			038D	900	
			038D	901	

```

MOVZBL #CJFS_RU,(SP)
JSB RMSWRTJNL_OBJ
MOVL R0,R6

MOVL R6,R0
ADDL2 #8,SP
POPR #^M<R4,R5,R6>
RSB

.END

```

```

; type of journal
; write jnl entry as OBJECT_ID
; save status code
;
; clean up stack
; restore work registers
; return to caller

```

RMOCRECOM
Symbol table

COMMON CREATE FILE

M 4

16-SEP-1984 00:15:06 VAX/VMS Macro V04-00
5-SEP-1984 16:21:31 [RMS.SRC]RMOCRECOM.MAR;1

Page 21
(11)

```

$$PSECT_EP          = 00000000
$$RMSTEST           = 0000001A
$$RMS_PBUGCHK       = 00000010
$$RMS_TBUGCHK       = 00000008
$$RMS_UMODE         = 00000004
ATRSC_EXPDATE       = 00000013
ATRSC_FPRO          = 00000016
ATRSC_RECATTR       = 00000004
ATRSC_UCHAR         = 00000003
ATRSC_UIC           = 00000015
ATRSL_ADDR          = 00000004
ATRSL_ENT_SIZE      = 00000008
ATRSL_TYPE          = 00000002
BDBSL_ADDR          = 00000018
BDBSW_NUMB          = 00000014
BR_AID              = 00001A5 R      01
CJFS_AI             = 00000003
CJFS_BI             = 00000002
CJFS_RU             = 00000001
CREXIT              = 00000060 R      01
CRE_XAB_ARGS        = 00000000 R      01
CRE_XAB_ARGS2       = 00000013 R      01
CTL$GL_RUF          = ***** X      01
DEL                 = 0000021C R      01
DELSHR              = 000001ED R      01
DEVSV_DIR            = 00000003
DEVSV_SQD           = 00000005
DOCIF               = 00000230 R      01
ERRALQ              = 00000061 R      01
ERRCREATE           = 00000265 R      01
ERRJOP              = 000001A0 R      01
ERRMOD              = 000001A7 R      01
EXIT                = 000001EC R      01
EXPIRE              = 00000317 R      01
FABSL_ALQ           = 00000010
FABSL_FOP           = 00000004
FABSM_TMD           = 00000010
FABSM_TMP           = 00000008
FABSV_CBT           = 00000015
FABSV_CIF           = 00000019
FABSV_CTG           = 00000014
FABSV_MXV           = 00000001
FABSV_SUP           = 00000002
FABSV_TMD           = 00000004
FABSV_UFO           = 00000011
FCHSV_CONTIG        = 00000007
FCHSV_CONTIGB       = 00000005
FIBSB_ALALIGN       = 00000021
FIBSC_LENGTH        = 00000040
FIBSC_RFI           = 00000004
FIBSL_EXSZ          = 00000018
FIBSL_EXVBN         = 0000001C
FIBSM_ALCON         = 00000001
FIBSM_FILCON        = 00000004
FIBSV_ALCONB        = 00000001
FIBSV_EXTEND        = 00000007
FIBSV_FINDFID       = 0000000B

```

```

FIBSV_HIGHVER       = 0000000F
FIBSV_LOWVER        = 0000000E
FIBSV_NEWVER        = 00000009
FIBSV_SUPERSEDE     = 0000000A
FIBSW_DID            = 0000000A
FIBSW_DID_SEQ       = 0000000C
FIBSW_EXCTL         = 00000016
FIBSW_LOC_FID       = 00000022
FIBSW_LOC_RVN       = 00000026
FIBSW_NMCTL         = 00000014
FILE_PRO             = 000002F0 R      01
FILE_UIC             = 000002FB R      01
FILNAM              = 000001CE R      01
FOP                  = 00000020
FWASL_ATR_WORK      = 00000058
FWASQ_FIB           = 00000010
FWAST_FIBBUF        = 000001F4
FWASW_UCHAR         = 00000044
GETJNC              = 00000132 R      01
IFBSB_JNLFLG        = 000000A0
IFBSB_ORGCASE       = 00000023
IFBSB_RFMORG        = 00000050
IFBSC_IDX           = 00000002
IFBSL_EBK           = 00000074
IFBSL_EBK_DISK      = 00000058
IFBSL_HBK           = 00000070
IFBSL_JNLBDB        = 00000030
IFBSL_PRIM_DEV      = 00000000
IFBSL_RJB           = 000000A4
IFBSM_NEVER_RU      = 00000020
IFBSM_ONLY_RU       = 00000001
IFBSM_RU            = 00000002
IFBSS_ORG           = 00000004
IFBSS_RFM           = 00000004
IFBSV_ORG           = 00000004
IFBSV_RFM           = 00000000
IFBSV_TMP           = 00000035
IMPSV_TEMP1         = 00000002
IOSM_ACCESS         = 00000040
IOSM_CREATE         = 00000080
IOSM_DELETE         = 00000100
IOSV_DELETE         = 00000008
IOS_ACCESS          = 00000032
IOS_CREATE          = 00000033
IOS_DELETE          = 00000035
IOS_MODIFY          = 00000036
JNLSC_ENT_SIZE      = 00000008
JNL$ACTION RTN      = 00000004
JNL$ATTR_TYPE       = 00000000
NAMS$FNB            = 00000034
NAMSV_HIGHVER       = 0000000F
NAMSV_LOWVER        = 0000000E
NAMCTC              = 00000108 R      01
RECORD_ATTR         = 00000306 R      01
RJB$V_AI            = 00000002
RJB$V_BI            = 00000001
RJB$V_RU            = 00000000

```


RMOCRECOM
Symbol table

COMMON CREATE FILE

N 4

16-SEP-1984 00:15:06 VAX/VMS Macro V04-00
5-SEP-1984 16:21:31 [RMS.SRC]RMOCRECOM.MAR;1

Page 22
(11)

RJBSW_FLAGS	= 0000000A		
RJRSB_ENTRY_TYPE	= 00000003		
RJRSB_FNS	= 00000058		
RJRSB_OPER	= 00000005		
RJRSB_ORG	= 00000004		
RJRSC_FILENAME	= 00000001		
RJRSC_FILNAMLEN	= 000001C4		
RJRSC_RECATRLEN	= 00000020		
RJRSL_ATR_FLAGS	= 0000003C		
RJRSL_PROT	= 00000044		
RJRSL_UCHAR	= 0000004C		
RJRSL_UIC	= 00000040		
RJRSG_EXPIRE	= 00000050		
RJRSS_FILENAME	= 00000100		
RJRST_C_FIB	= 00000064		
RJRST_FILENAME	= 000000C4		
RJRST_REC_ATTR	= 000000A4		
RJRSV_ATR_EXPIRE	= 00000004		
RJRSV_ATR_PROT	= 00000001		
RJRSV_ATR_REC	= 00000003		
RJRSV_ATR_UCHAR	= 00000000		
RJRSV_ATR_UIC	= 00000002		
RJRSC_CREATE	= 00000004		
RMSACLOC_RJB_BDB	*****	X	01
RMSASSJNL	*****	X	01
RMSCLSCU	*****	X	01
RMSCREACC_SET1	*****	X	01
RMSCREACC_SET2	*****	X	01
RMSCREACC_SET3	*****	X	01
RMSCREATECOM	0000004A	RG	01
RMSFCPFNC	*****	X	01
RMSFCPFNC_P4	*****	X	01
RMSFCP_P4_P2	*****	X	01
RMSFILNAM	*****	X	01
RMSGETFILNAM	*****	X	01
RMSGETJNL	*****	X	01
RMSGETSPC1	*****	X	01
RMSINI_CRE_RJR	0000026D	R	01
RMSJNL_CREATE	00000322	R	01
RMSMAPERR	*****	X	01
RMSOPEN_CIF	*****	X	01
RMSRETSPC1	*****	X	01
RMSRTVJNL	*****	X	01
RMSSETDID	*****	X	01
RMSSETEBK	*****	X	01
RMSSETHBK	*****	X	01
RMSWRTJNL_OBJ	*****	X	01
RMSXAB_SCAN	*****	X	01
RMS\$_ACQ	= 00018404		
RMS\$_CRE	= 0001C00A		
RMS\$_JOP	= 000187E4		
RMS\$_NRU	= 000187FC		
RUCBSB_CTRL	= 00000011		
RUCBSV_ACTIVE	= 00000001		
SET3	0000011A	R	01
SETEBK	000001C1	R	01
SETHBK	000001B1	R	01

SETHBK_BR	0000019E	R	01
SETTMP	00000067	R	01
SETUP	0000006B	R	01
SS\$ NORMAL	= 00000001		
TAB\$C_EXPIRE	= 00000004		
TAB\$C_FILE_PRO	= 00000001		
TAB\$C_FILE_UIC	= 00000002		
TAB\$C_RECORD_ATTR	= 00000003		
TAB\$C_USER_CHAR	= 00000000		
TABLE	0000001A	R	01
USER_CHAR	000002E5	R	01
XAB\$C_ALL	= 00000014		
XAB\$C_ALLLEN	= 00000020		
XAB\$C_ANY	= 00000000		
XAB\$C_DAT	= 00000012		
XAB\$C_DATLEN_V2	= 00000024		
XAB\$C_FHC	= 0000001D		
XAB\$C_FHCLEN	= 0000002C		
XAB\$C_JNL	= 00000022		
XAB\$C_JNLLEN	= 0000003C		
XAB\$C_PRO	= 00000013		
XAB\$C_PROLEN_V3	= 00000010		
XAB\$C_RDT	= 0000001E		
XAB\$C_RDTLEN	= 00000014		
XBC\$C-CREALL1	*****	X	01
XBC\$C-CREALL2	*****	X	01
XBC\$C-CREJNL	*****	X	01
XBC\$C-CREPRO	*****	X	01
XBC\$C-CREPRO1	*****	X	01
XBC\$C-OPNDAT	*****	X	01
XBC\$C-OPNFHC	*****	X	01
XBC\$C-OPNRDT	*****	X	01
XITPOP	000001EA	R	01

! Psect synopsis !

PSECT name	Allocation	PSECT No.	Attributes															
ABS	00000000 (0.)	00 (0.)	NOPIC	USR	CON	ABS	LCL	NOSHR	NOEXE	NORD	NOWRT	NOVEC	BYTE					
RM\$RMS0	0000038D (909.)	01 (1.)	PIC	USR	CON	REL	GBL	NOSHR	EXE	RD	NOWRT	NOVEC	BYTE					
\$AB\$\$	00000000 (0.)	02 (2.)	NOPIC	USR	CON	ABS	LCL	NOSHR	EXE	RD	WRT	NOVEC	BYTE					

! Performance indicators !

Phase	Page faults	CPU Time	Elapsed Time
Initialization	36	00:00:00.09	00:00:00.63
Command processing	129	00:00:00.75	00:00:05.48
Pass 1	628	00:00:27.26	00:01:18.43
Symbol table sort	0	00:00:04.53	00:00:09.44
Pass 2	163	00:00:05.05	00:00:13.30
Symbol table output	24	00:00:00.23	00:00:01.21
Psect synopsis output	2	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	984	00:00:37.93	00:01:48.54

The working set limit was 1950 pages.
154836 bytes (303 pages) of virtual memory were used to buffer the intermediate code.
There were 160 pages of symbol table space allocated to hold 3042 non-local and 38 local symbols.
901 source lines were read in Pass 1, producing 17 object records in Pass 2.
40 pages of virtual memory were used to define 39 macros.

! Macro library statistics !

Macro library name	Macros defined
_\$255\$DUA28:[RMS.OBJ]RMS.MLB;1	20
_\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	5
_\$255\$DUA28:[SYSLIB]STARLET.MLB;2	10
TOTALS (all libraries)	35

3181 GETs were required to define 35 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LISS:RMOCRECOM/OBJ=OBJ\$:RMOCRECOM MSRC\$:RMOCRECOM/UPDATE=(ENH\$:RMOCRECOM)+EXECML\$/LIB+LIB\$:RMS/LIB

0318 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

